Distributed prediction of time series data with kernels and adaptive filtering techniques in sensor networks

Paul Honeine[†], Cédric Richard[†], José Carlos M. Bermudez[§] and Hichem Snoussi[†]
[†] Institut Charles Delaunay (ICD, FRE CNRS 2848), Laboratoire LM2S
Université de Technologie de Troyes, BP 2060, 10010 Troyes - France
{paul.honeine, cedric.richard, hichem.snoussi}@utt.fr
[§] Department of Electrical Engineering
Federal University of Santa Catarina 88040-900, Florianópolis, SC - Brazil
j.bermudez@ieee.org

***Abstract*—Wireless sensor networks are becoming versatile tools for learning a physical phenomenon, monitoring its variations and predicting its evolution. They rely on low-cost tiny devices which are deployed in the region under scrutiny and collaborate with each other. Limited computation and communication resources require special care in designing distributed prediction algorithms for sensor networks. In this communication, we propose a nonlinear prediction technique that takes advantage of recent developments in kernel machines and adaptive filtering for online nonlinear functional learning. Conventional methods, however, are inappropriate for large-scale sensor networks, as the resulting model corresponds to the number of deployed sensors. To circumvent these drawbacks, we consider a distributed control of the model order. The model parameters are transmitted from sensor to sensor and updated by each sensor based the measurement information. The model order is incremented whenever this increment is relevant compared to a reduced-order model. The proposed approach is naturally adapted for predicting a varying phenomenon, as model order increases are governed by the novelty of the new observation at each sensor node. We illustrate the applicability of the proposed technique by some simulations on establishing the temperature map in an region heated by sources.**

I. Introduction

Wireless sensor networks (WSN) are becoming a multidisciplinary research area attracting researchers from microelectronics, communication and signal processing communities, only to name a few. Mainly motivated by military applications, they are appropriate for a broad class of applications, including monitoring physical or environmental conditions, such as temperature, humidity or pollutants for supervising an agricultural field at different locations. They rely on tiny devices with sensing capabilities, communicating wirelessly with each other. In order to cover a large spatial region of interest, a big number of these devices (simply denoted as sensors hereafter) are deployed randomly, restricting the manufacturing cost of each one. This leads to resource-constraint sensors, limiting the complexity of the algorithm run on each one.

Sensors are mainly deployed to get some measurements at known (random) locations[1] in the region under scrutiny. Since communication between sensors is possible, at least between neighbors, they may collaborate with each other to learn the overall physical phenomenon in each location in the region, from the measurements as well as the locations of the sensors. As opposed to a fusion center approach where sensors only sense and convey information, here we have *smart* sensors. A common learning scheme consists of updating the model, at each sensor in turn, with the newly acquired data, and forwards information to another nearby sensor.

[1]Even when deployed randomly, we assume that each sensor is location-aware, with coordinates obtained by a self-localization pre-processing technique, as illustrated in [1] and references therein. Moreover, most applications involve motionless devices.

One can take advantage of adaptive learning, as preconized by the vast literature on adaptive filtering techniques [2], [3]. However, most of these are linear signal processing techniques, inappropriate for the inherent nonlinear nature of the wide class of systems under investigation. Nonlinear techniques are becoming more widely investigated thanks to recent developments in machine learning, and more specifically with kernel machines; See for instance [4] for a review of kernel methods. Most of these techniques are nonlinear analysis and processing techniques obtained by revisiting linear ones in the light of reproducing kernel Hilbert spaces. There is a growing interest around this concept for data mining and machine intelligence, since recent burst of Vapnik's support vector machines and statistical learning theory [5]. While most kernel machines yield optimal solutions, they are not established in the same line as online learning, as the order of the resulting model grows along with the number of available observations.

In this paper, we propose a nonlinear kernel-based adaptive method, applied here for WSN, by bridging the gap between the nonlinear kernel machines and adaptive filtering techniques. In [6], [7], we have proposed to control the order of the resulting model, with the pre-processing kernel-coherence criterion, and revisited in [8] for learning with WSN. Nevertheless, such criterion is used prior to any learning scheme, and is independent of the prediction error. Here, we revisit this criterion to propose a post-processing criterion, the functional-coherence criterion, which is applied to the learnt function, thus taking into account the prediction error. The rest of this paper is organized as follows. In section II, learning in a RKHS is briefly introduced, with focus on functional leaning. In section III, we derive techniques to control the order. Then the proposed method is studied in section IV. Simulation results are reported in section V, and finally, conclusion remarks are summarized in section VI.

II. Functional learning with kernel machines

In order to understand the nature of functional learning with kernel machines, it will be useful first to take a look at a conventional least-squares problem for fitting data. Let $(\boldsymbol{x}_1, d_1), (\boldsymbol{x}_2, d_2), \ldots, (\boldsymbol{x}_n, d_n)$ be a set of n training pairs, where $\boldsymbol{x}_i$ is an input example drawn from a vector-space $\mathcal{X}$, and $d_i \in \mathcal{D}$ is the desired observation or label associated to it. The goal is to learn a map from $\mathcal{X}$ to $\mathcal{D}$ given the training set, and therefore getting the appropriate d for any $\boldsymbol{x}$. Mathematically, the mapping or system *connecting* the two spaces is modeled by a function, say ψ. When the labels take values in a finite set, the task is called classification or discrimination. In a more general setting[2], when $\mathcal{D} = \mathbb{R}$ we have a regression problem where d_i corresponds to the desired output of the system ψ for the input $\boldsymbol{x}_i$. The optimal function ψ^* is obtained by minimizing some cost functional on the available training set, such as the mean square error, by solving the optimization problem

$$\psi^* = \arg\min_{\psi} \sum_{i=1}^{n} |\psi(\boldsymbol{x}_i) - d_i|^2. \tag{1}$$

However, this is an ill-posed problem, as an infinite number of functions sets to zero this cost functional. In order to obtain a well-posed problem, one must restrict the space of candidate functions into for instance linear, quadratic, or polynomial functions or even splines.

Linear functions are conceptually easy and simple to implement, as preconized by the large literature on adaptive filtering techniques [2], [3]. For this purpose, we restrict ourselves to functions ψ defined by a vector $\boldsymbol{a} \in \mathcal{X}$ such that $\psi(\boldsymbol{x}) = \langle \boldsymbol{a}, \boldsymbol{x} \rangle$ for any $\boldsymbol{x} \in \mathcal{X}$, where $\langle \cdot, \cdot \rangle$ is the usual inner product in a vector space. Thus, the optimization problem is reduced to determine the vector $\boldsymbol{a}^*$ by solving

$$\boldsymbol{a}^* = \arg\min_{\boldsymbol{a}} \sum_{i=1}^{n} \big|\langle \boldsymbol{a}, \boldsymbol{x}_i \rangle - d_i\big|^2, \tag{2}$$

This optimal vector can be learnt recursively in an online learning scheme as suggested with adaptive filtering techniques, where at each time-instance n we have a new observation pair $(\boldsymbol{x}_n, d_n)$. Nevertheless, this optimization problem may still be ill-posed, and some structure of the candidates vectors $\boldsymbol{a}$ must be required, taking into account for instance the regularity of the system as well as the noisy measurements. Moreover, it is worth noting that, by their nature, such linear functions are not suitable to nonlinear analysis and prediction.

In this paper, we take advantage of recent developments in the area of nonlinear functional learning, with kernel machines. The space of candidate functions is constructed on the underlying structure of reproducing kernel functions, as defined next. Let $(\mathcal{H}, \langle \cdot\ , \cdot \rangle_{\mathcal{H}})$ denotes a Hilbert space of real-valued functions defined on $\mathcal{X}$. A reproducing kernel of $\mathcal{H}$ is a function $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ from $\mathcal{X} \times \mathcal{X}$ to $\mathbb{R}$ that verifies both properties:

- the function $\kappa(\boldsymbol{x}_i, \cdot)$, belongs to $\mathcal{H}$ for all $\boldsymbol{x}_i \in \mathcal{X}$,
- $\psi(\boldsymbol{x}_i) = \langle \kappa(\boldsymbol{x}_i, \cdot), \psi \rangle_{\mathcal{H}}$ for all $\boldsymbol{x}_i \in \mathcal{X}$ and $\psi \in \mathcal{H}$.

The Hilbert space $\mathcal{H}$ associated with this kernel is called a reproducing kernel Hilbert space (RKHS). The last property is a fundamental property of RKHS, which states that the evaluation of a function $\psi \in \mathcal{H}$ in that space at any $\boldsymbol{x}_i \in \mathcal{X}$ is given by its inner product with the function $\kappa(\boldsymbol{x}_i, \cdot)$. Moreover, one can write for the particular case $\psi = \kappa(\boldsymbol{x}_j, \cdot)$ for any $\boldsymbol{x}_j \in \mathcal{X}$ the property $\langle \kappa(\boldsymbol{x}_i, \cdot), \kappa(\boldsymbol{x}_j, \cdot) \rangle_{\mathcal{H}} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$. Therefore, $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ corresponds to an inner product between data in a *modified* or *transformed* space. Back to the optimization problem (1) and its linear counterpart (2) applied to the RKHS, we get the following optimization problem

$$\psi^* = \arg\min_{\psi \in \mathcal{H}} \sum_{i=1}^{n} |\psi(\boldsymbol{x}_i) - d_i|^2 + \eta \|\psi\|^2_{\mathcal{H}}. \tag{3}$$

In this expression, we use a Tikhonov regularization with $\eta \in [0, 1]$, allowing a tradeoff between the well-fitness to the data (the first term in the above cost function) and the regularity of the resulting function (the second term). It is worth noting that we get a linear-function problem (2) by considering the linear reproducing kernel $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle$. Nonlinearity is achieved by substituting the inner product by a nonlinear reproducing kernel as defined in Table I, leaving the algorithm unchanged and incurring essentially the same computational cost.

Polynomial	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle + \beta^2)^p$
Laplace	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|/\sigma_0^2\right)$
Gaussian	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/2\sigma_0^2\right)$

TABLE I
Some reproducing kernels, with tunable parameters β, p, σ_0.

III. Model order control for online learning

By virtue of the well known Representer Theorem [9], [10], solutions to the regularized optimization problem (3) have the form

$$\psi^* = \sum_{j=1}^{n} \alpha_j^*\, \kappa(\boldsymbol{x}_j, \cdot), \tag{4}$$

where the order of the model corresponds to the number of available data. By analogy to the linear case defined in (2), $\boldsymbol{x}_i$ are substituted here by their corresponding kernel functions $\kappa(\boldsymbol{x}_i, \cdot)$, and $\boldsymbol{a}^*$ by the function ψ^* defined in (4). Inserting this model into (3) leads to the following optimization problem:

$$\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha}} \|\boldsymbol{d} - \boldsymbol{K}\boldsymbol{\alpha}\|^2 + \eta\, \boldsymbol{\alpha}^\top \boldsymbol{K} \boldsymbol{\alpha},$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{d}$ are n-dimensional column vectors of entries α_k and d_k, respectively, and $\boldsymbol{K}$ is the n-by-n (Gram) matrix of entries $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$, for $i, j = 1, \ldots, n$. From classical matrix algebra, we get the solution to this well known problem with

$$(\boldsymbol{K} + \eta \boldsymbol{I})\, \boldsymbol{\alpha}^* = \boldsymbol{d},$$

[2]An even more general setting consists of $\mathcal{D} = \mathbb{R}^d$ or even $\mathcal{D} = \mathbb{C}^d$, however such straightforward generalization is beyond the scope of this communication.

with $\boldsymbol{I}$ the identity matrix of appropriate size. Therefore, this requires the inversion of the matrix $\boldsymbol{K} + \eta \boldsymbol{I}$, which is warrantable thanks to a non-zero regularization constant η.

In an online setting, we have a new data pair $(\boldsymbol{x}_n, d_n)$ at each instant n, and the model is updated recursively based on this new information. However, considering a model of the form (4), its order will steadily increasing. In order to circumvent this inconvenience, we propose to control its order. This is done by retaining only a small number of kernel functions in the expansion (4), say m, and write

$$\psi_n = \sum_{k=1}^{m} \alpha_{n,k}\, \kappa(\boldsymbol{x}_{\omega_k}, \cdot), \tag{5}$$

where $\omega_1, \ldots, \omega_m$ are the indexes of the retained kernel functions, from the n available ones. The selection criterion determines if, at instant n, the kernel function $\kappa(\boldsymbol{x}_n, \cdot)$ should be included to the expansion, or discarded, or even removed if it already belongs to it. Therefore, the model order may increment, stay unchanged, or decrement. In [6], [8], [7], we derived an easy to compute criterion applied in a pre-processing scheme, independently of the resulting optimal function as well as the prediction error. In order to circumvent these drawbacks, we investigate in this paper a new criterion applied in a post-processing scheme, after getting the optimal function and therefore is observation-aware as opposed to the former. Since this criterion follows the same spirit of the former, we begin by a brief review of the former criterion.

A. The kernel-coherence criterion, a pre-processing criterion

In this first approach, we seek a compact representation, where the current kernel function is discarded from the model expansion when it can be well approximated by the already available ones. This can be done by comparing $\kappa(\boldsymbol{x}_n, \cdot)$ to its projection onto the space spanned by the other m kernel functions, yielding the following optimization problem

$$\min_{\boldsymbol{\gamma}} \|\kappa(\boldsymbol{x}_n, \cdot) - \sum_{j=1}^{m} \gamma_j\, \kappa(\boldsymbol{x}_{\omega_j}, \cdot)\|_{\mathcal{H}}^2 < \nu_0^2, \tag{6}$$

for a given threshold ν_0. From matrix algebra, this is solved by an inversion of the m-by-m Gram matrix of entries $\kappa(\boldsymbol{x}_{\omega_i}, \boldsymbol{x}_{\omega_j})$ for $i, j = 1, \ldots, m$, as given for instance in [11], and thus has a computational complexity of $\mathcal{O}(m^3)$. In [6], [7], we consider the following more efficient criterion. We discard $\kappa(\boldsymbol{x}_n, \cdot)$ from the expansion if, for a given threshold ν_1, we have

$$\max_{j=1,\ldots,m} \frac{|\langle \kappa(\boldsymbol{x}_n, \cdot), \kappa(\boldsymbol{x}_{\omega_j}, \cdot)\rangle_{\mathcal{H}}|}{\|\kappa(\boldsymbol{x}_n, \cdot)\|_{\mathcal{H}} \|\kappa(\boldsymbol{x}_{\omega_j}, \cdot)\|_{\mathcal{H}}} > \nu_1. \tag{7}$$

In other terms, this corresponds to upper-bounding the cosine of the angles between the kernel functions in the resulting expansion. For a small threshold value, we get a set of incoherent kernel functions, and orthogonality when $\nu_1 = 0$; we call this criterion the kernel-coherence criterion. This criterion is very efficient from a computational point of view, since the numerator of (7) is given by $|\kappa(\boldsymbol{x}_n, \boldsymbol{x}_{\omega_j})|$ and its denominator is $\sqrt{\kappa(\boldsymbol{x}_n, \boldsymbol{x}_n)\kappa(\boldsymbol{x}_{\omega_j}, \boldsymbol{x}_{\omega_j})}$.

In [7], we derive the connection between both criteria (6) and (7). These criteria consider the most compact representation (9) that approximates well the optimal model (4). However, they do not take into account the observation d_n associated to $\kappa(\boldsymbol{x}_n, \cdot)$, neither the prediction error. This is mainly due to the pre-processing nature of the approach, as selection occurs before any computation of the resulting model, with and without the kernel function $\kappa(\boldsymbol{x}_n, \cdot)$.

B. The functional-coherence criterion, a post-processing criterion

The kernel-coherence criterion mentioned above, as defined in (7), determines the relevance of the information in $\kappa(\boldsymbol{x}_n, \cdot)$, with respect to all previous kernel functions in the expansion. We propose to revisit this principle, and apply it to the learnt function. For now, consider that the function ψ_n is computed with an expansion including $\kappa(\boldsymbol{x}_n, \cdot)$. We propose to reduce the model order by discarding this kernel function if ψ_n is not very relevant, compared to previous computed functions, with

$$\max_{k=1,\ldots,n-1} \frac{\langle \psi_n, \psi_k \rangle_{\mathcal{H}}}{\|\psi_n\|_{\mathcal{H}} \|\psi_k\|_{\mathcal{H}}} > \nu_0',$$

where ψ_k is the optimal function obtained at instant k. This criterion may be considered as an extension of (7), and in the same line of upper-bounding the cosine of the angles between functions in order to *span* the largest subspace in the RKHS. As opposed to (7), this criterion considers the whole learnt function, including its prediction error, and not only its kernel expansion. While this requires keeping all previously learnt functions, we derive a sufficient condition by considering the criterion

$$\frac{\langle \psi_n, \psi_n^{\perp} \rangle_{\mathcal{H}}}{\|\psi_n\|_{\mathcal{H}} \|\psi_n^{\perp}\|_{\mathcal{H}}} > \nu_1',$$

for a given threshold $\nu_1' \in [0, 1]$, and where $\psi_n^{\perp}$ is the orthogonal projection of ψ_n onto the space spanned by the $n-1$ functions. Moreover, we don't need to keep all these functions in memory, since this space is also spanned by the retained kernel functions. In the next section, we propose a learning algorithm, where this criterion is applied at the end in order to determine which of ψ_n or $\psi_n^{\perp}$ must be considered as the resulting optimal function, based on this measure of coherence between them.

IV. Distributed functional learning in WSN

Consider the problem of functional learning in wireless sensor networks (WSN), such as estimating a temperature field for instance, where the input of the function is the coordinates in the region under scrutiny, and its output is estimated temperature. We learn the optimal function from sensors deployed in this region, giving us measurements at known locations. Since most considered techniques are iterative, this paper is not an exception, more than one pass through the network may be required in order to converge to the optimal solution, as well as in case of tracking the evolution of the system.

In [8], we derived a distributed regression technique based on the kernel-coherence criterion. However, this criterion depends only on the location of the sensors, $\boldsymbol{x}_n$, thus the model is fixed for a given deployment for a the large class of motionless-sensor problem (at least after the first pass through the network), and only the weighting coefficients are updated. The functional-coherence criterion allows us to overcome this drawback, since it looks at the functions where the prediction error is considered in the iterative updating scheme. This approach is illustrated in this paper, with a kernel-based normalized least-mean-squares (NLMS) technique derived next.

Consider the optimization problem (3), where ψ is substituted by the reduced-order model (9). Each sensor, in turn, updates the model ψ_n with the newly acquired data d_n and forward information to another nearby sensor, and so on. The following actions are performed:

A. An updating parameter stage, with or without changing the model order depending on whether $\kappa(\boldsymbol{x}_n, \cdot)$ does not belong to the the model expansion, or it already belongs. The model resulting from this stage, ψ_n incorporates this kernel function. Let $m+1$ be its order.
B. A projection stage, where we compute $\psi_n^\top$, the projection of ψ_n onto the space with one removed kernel function, $\kappa(\boldsymbol{x}_n, \cdot)$. The resulting model, $\psi_n^\perp$ does not include the latter. Its order is m.
C. A model order reduction stage, by considering as a final model $\psi_n^\top$ rather than ψ_n, if

$$\frac{\langle \psi_n, \psi_n^\perp \rangle_{\mathcal{H}}}{\|\psi_n\|_{\mathcal{H}} \|\psi_n^\perp\|_{\mathcal{H}}} > \nu_1'.$$

Next, we develop each of these stages, while the first stage is decomposed into two cases, depending if the current kernel function already belongs to the model, or not. For the sake of simplicity, we begin by developing the algorithm for $\eta = 0$ in both cases, and introduce the regularization parameter at the end of each.

A1. *if $\kappa(\boldsymbol{x}_n, \cdot)$ already belongs to the model*

Without changing the model order, defined by $\psi_n = \sum_{k=1}^{m+1} \alpha_{n,k}\, \kappa(\boldsymbol{x}_{\omega_k}, \cdot)$, we consider updating the coefficients, $\alpha_{n,k}$, by solving the following minimum-deviation problem

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \|\boldsymbol{\alpha} - \boldsymbol{\alpha}_{n-1}\|^2 \\ \text{subject to} \quad & \boldsymbol{\alpha}^\top \boldsymbol{\kappa}_n = d_n, \end{aligned}$$

where $\boldsymbol{\kappa}_n = \big[\kappa(\boldsymbol{x}_{\omega_1}, \boldsymbol{x}_n) \cdots \kappa(\boldsymbol{x}_{\omega_{m+1}}, \boldsymbol{x}_n)\big]^\top$, and the resulting $\boldsymbol{\alpha}_n$ is a column vector of entries $\alpha_{n,k}$ for $k = 1, \ldots, m+1$. This constraint is equivalent to $\psi_n(\boldsymbol{x}_n) = d_n$. To solve the constraint optimization problem, we write the Lagrangian

$$\|\boldsymbol{\alpha} - \boldsymbol{\alpha}_{n-1}\|^2 + \lambda(d_n - \boldsymbol{\alpha}^\top \boldsymbol{\kappa}_n)$$

where λ denotes the Lagrangian multiplier. By taking to zero its derivatives with respect to $\boldsymbol{\alpha}$ and λ, we get the two conditions on $\boldsymbol{\alpha}_n$:

$$\boldsymbol{\alpha}_n - \boldsymbol{\alpha}_{n-1} = \frac{1}{2}\lambda\, \boldsymbol{\kappa}_n^\top \qquad \text{and} \qquad \boldsymbol{\alpha}_n^\top \boldsymbol{\kappa}_n = d_n$$

From these conditions on $\boldsymbol{\alpha}_{n-1}$, we get $\lambda = 2(\boldsymbol{\kappa}_n^\top \boldsymbol{\kappa}_n)^{-1}(d_n - \boldsymbol{\alpha}_{n-1}^\top \boldsymbol{\kappa}_n)$, where $\boldsymbol{\kappa}_n^\top \boldsymbol{\kappa}_n$ is assumed non-singular. By injecting this expression in the first condition, we get the recursive update

$$\boldsymbol{\alpha}_n = \boldsymbol{\alpha}_{n-1} + \frac{\rho}{\eta + \|\boldsymbol{\kappa}_n\|^2}\, \boldsymbol{\kappa}_n (d_n - \boldsymbol{\alpha}_{n-1}^\top \boldsymbol{\kappa}_n), \tag{8}$$

where ρ is step-size control parameter, preconised in the adaptive filtering literature, and where we reintroduce the regularization parameter η. In this expression, $d_n - \boldsymbol{\alpha}_{n-1}^\top \boldsymbol{\kappa}_n$ corresponds to the prediction error resulting from the previous model.

A2. *if $\kappa(\boldsymbol{x}_n, \cdot)$ does not belong to the model*

In this case, we increment the model order by including this kernel function to the expansion, yielding the model

$$\psi_n = \sum_{k=1}^{m} \alpha_{n,k}\, \kappa(\boldsymbol{x}_{\omega_k}, \cdot) + \alpha_{n,m+1}\, \kappa(\boldsymbol{x}_n, \cdot). \tag{9}$$

With one more coefficient and kernel function, the minimum-deviation problem becomes

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \|\boldsymbol{\alpha}_{[1:m]} - \boldsymbol{\alpha}_{n-1}\|^2 + \alpha_{m+1}^2 \\ \text{subject to} \quad & \boldsymbol{\alpha}^\top \boldsymbol{\kappa}_n = d_n, \end{aligned}$$

where $\boldsymbol{\alpha}_{[1:m]}$ denotes the first m entries of vector $\boldsymbol{\alpha} = [\alpha_1 \cdots \alpha_m \quad \alpha_{m+1}]^\top$, and $\boldsymbol{\kappa}_n = [\kappa(\boldsymbol{x}_{\omega_1}, \boldsymbol{x}_n) \cdots \kappa(\boldsymbol{x}_{\omega_m}, \boldsymbol{x}_n) \quad \kappa(\boldsymbol{x}_n, \boldsymbol{x}_n)]^\top$. Similarly to A1, we get the recursive updating

$$\boldsymbol{\alpha}_n = \begin{bmatrix} \boldsymbol{\alpha}_{n-1} \\ 0 \end{bmatrix} + \frac{\rho}{\eta + \|\boldsymbol{\kappa}_n\|^2}\, \boldsymbol{\kappa}_n \Big(d_n - \boldsymbol{\kappa}_n^\top \begin{bmatrix} \boldsymbol{\alpha}_{n-1} \\ 0 \end{bmatrix}\Big). \tag{10}$$

B. *Projection stage*

In order to simplify the notations, let $\kappa(\boldsymbol{x}_{\omega_{m+1}}, \cdot)$ be the current kernel function, i.e. $n = \omega_{m+1}$ Let $\psi_n^\perp$ be the orthogonal projection of ψ_n onto the space spanned by the other m kernel functions, obtained by solving

$$\langle \kappa(\boldsymbol{x}_{\omega_j}, \cdot), \psi_n - \psi_n^\perp \rangle_{\mathcal{H}} = 0 \qquad \text{for all} \qquad j = 1, \ldots, m.$$

Since we have $\psi_n^\perp = \sum_{j=1}^m \beta_{n,j}\, \kappa(\boldsymbol{x}_{\omega_j}, \cdot)$, this is done by solving the m-by-m linear system

$$\boldsymbol{K}_\omega\, \boldsymbol{\beta}_n = \boldsymbol{\psi}_\omega, \tag{11}$$

where $\boldsymbol{\beta}_n$ and $\boldsymbol{\psi}_\omega$ are m-element column vectors of entries $\beta_{n,j}$ and $\psi_n(\boldsymbol{x}_{\omega_j})$ for $j = 1, \ldots, m$, respectively, and $\boldsymbol{K}_\omega$ is the m-by-m matrix of entries $\kappa(\boldsymbol{x}_{\omega_i}, \boldsymbol{x}_{\omega_j})$ for $i, j = 1, \ldots, m$. It is worth noting that we can make use of a rank one update for the matrix inversion process.

C. *Model order reduction stage*

In both cases, A1. and A2., the resulting model includes the kernel function $\kappa(\boldsymbol{x}_n, \cdot)$. Now we are in a position to reduce the model order, by using $\psi_n^\perp$ instead of ψ_n if

$$\frac{\langle \psi_n, \psi_n^\perp \rangle_{\mathcal{H}}}{\|\psi_n\|_{\mathcal{H}} \|\psi_n^\perp\|_{\mathcal{H}}} > \nu_1';$$

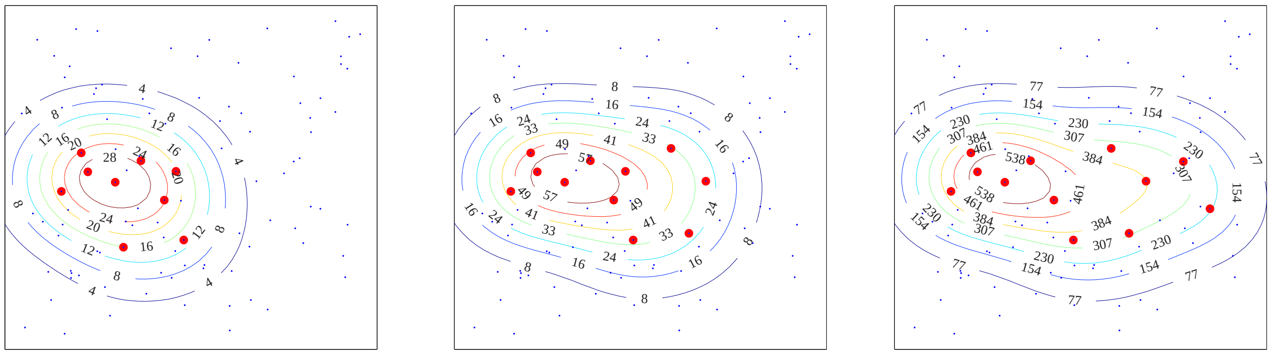

Fig. 1. The evolution of the model at $t = 1$, $t = 2$, and $t = 20$, with contour lines. The blue dots and the big-red dots represent the sensors and those contributing to the model, respectively.

otherwise, we keep ψ_n as the final model. Let $\boldsymbol{\alpha}_n$ be the column vector of the weighting coefficients, obtained from (8) or (10). Then, this criterion is equivalent to

$$\frac{(\boldsymbol{\alpha}_n^\top \boldsymbol{\psi}_\omega)^2}{\boldsymbol{\alpha}_n^\top \boldsymbol{K}_\omega \boldsymbol{\alpha}_n \; \boldsymbol{\psi}_\omega^\top \boldsymbol{K}_\omega^{-1} \boldsymbol{\psi}_\omega} > \nu_1'^2,$$

where relation (11) is used.

V. Simulations

In order to illustrate our approach, we consider the problem of monitoring diffusion phenomena governed by the generic partial differential equation

$$\frac{\partial C(\boldsymbol{x}, t)}{\partial t} - \nabla(D\,\nabla C(\boldsymbol{x}, t)) = Q(\boldsymbol{x}, t)$$

with C the concentration as a function of location and time, Q a volume source, which may depend on both location and time, and the diffusion parameter D (scalar or matrix), depending on the medium. When D is a constant, the diffusion is isotropic, and when D varies with direction, it is called anisotropic. In this paper, we consider an anisotropic medium[3], with several concrete examples such as studying the behavior of a pollutant concentration in the air in a windy environment, or monitoring a contaminating agent in water resources where the water flow takes part in its spread, in a given direction.

For this purpose, we consider 100 sensors in a fixed 2D location, deployed randomly in the 1.5-by-1.5 square region under scrutiny. We study the evolution of the system from $t = 1$ to $t = 20$, where at each instance we have one new measurement per sensor. From results on preliminary experiments done on instant $t = 10$, with 10 passes, we set $\eta = 0$, $\rho = 0.99$ and $\nu_1' = 0.99$. The commonly investigated Gaussian kernel is used, $\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/2\sigma^2)$, with its bandwidth set to $\sigma = 0.2$.

Figure 1 illustrates the evolution of the resulting model, at different time instances. We see that at time instant $t = 1$, we have a concentration within a given region. While the diffusion operates mainly in the x-direction, our model follows this anisotropic diffusion. This is illustrated by the contour lines in Figure 1, as well as with the (big-red) selected sensors defining the model, spanning the region under evolution

[3]For simulations, we use the PDE toolbox of Matlab, with D a function increasing with the x-coordinate, and Q a time-increasing source.

VI. Conclusion

This paper introduces a new criterion to control the model order of kernel machines. Compared to previously proposed criterion which are applied prior to any learning task and independent of the observations, this functional-coherence criterion operates on the learnt function. It can be regarded as a measure of coherence between the learnt function and it projection onto a smaller-dimension space, which determines if we keep the function or its projection as the final result.

References

[1] P. Honeine, C. Richard, M. Essoloh, and H. Snoussi, "Localization in sensor networks - a matrix regression approach," in 5th *IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM)*, Darmstadt, Germany, July 2008.

[2] S. Haykin, *Adaptive filtering theory*, Prentice Hall, NJ, USA, fourth edition edition, 2002.

[3] A.H. Sayed, *Fundamentals of adaptive filtering*, Wiley-IEEE Press, NY, USA, June 2003.

[4] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.

[5] V. Vapnik, *Statistical Learning Theory*, Wiley, New York, NY, USA, September 1998.

[6] P. Honeine, C. Richard, and J. C. M. Bermudez, "On-line nonlinear sparse approximation of functions," in *Proc. IEEE International Symposium on Information Theory (ISIT)*, Nice, France, June 2007.

[7] C. Richard, J. C. M. Bermudez, and P. Honeine, "Online prediction of time series data with kernels," *accepted in IEEE Trans. Signal Processing*, 2009.

[8] P. Honeine, M. Essoloh, C. Richard, and H. Snoussi, "Distributed regression in sensor networks with a reduced-order kernel model," in *IEEE Globecom'08*, New Orleans, LA, USA, 2008.

[9] G. Kimeldorf and G. Wahba, "Some results on tchebycheffian spline functions," *Journal of Mathematical Analysis and Applications*, vol. 33, pp. 82–95, 1971.

[10] B. Schölkopf, R. Herbrich, and R. Williamson, "A generalized representer theorem," Tech. Rep. NC2-TR-2000-81, Royal Holloway College, Univ. of London, UK, 2000.

[11] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least squares algorithm," *IEEE Trans. Signal Processing*, vol. 52, no. 8, pp. 2275–2285, 2004.